

Semi-generic analysis of Kraken results from Sunbeam

Lev Litichevskiy

October 12, 2021

Required input:

- Kraken2 output from Sunbeam (`demo_kraken_all_samples.tsv`)
- Sample metadata (could also construct it from the sample names in the Kraken dataset)

Analyses we will perform:

- Alpha diversity
- Barplots of microbial composition
- Beta diversity AKA PCoA
- DESeq2 to identify differential taxa

Search the document for “CHANGEME” to see which lines of code will require project-specific tweaks.

Load libraries

```
library(tidyverse)
library(phyloseq)
library(DESeq2)

# speedyseq is just to speed up tax_glom
# remotes::install_github("mikemc/speedyseq")
library(speedyseq)

theme_set(theme_bw(base_size=15))
```

Import data

```
# CHANGEME: change filepath to your input data
# N.B. You might need to delete the "#" on the second line (right before "OTU ID")
# in your input file. Do this in any text editor, like Notepad
kraken.df <- read.table("data/demo_kraken_all_samples.tsv",
                        sep="\t", quote="", header=T)

# remove "-taxa" suffix
colnames(kraken.df) <- str_replace(colnames(kraken.df), ".taxa$", "")

# subset to just data columns
kraken.data.df <- kraken.df %>%
  column_to_rownames("OTU.ID") %>%
  dplyr::select(-Consensus.Lineage)
```

```
dim(kraken.data.df)
```

```
## [1] 6565 16
```

```
head(kraken.data.df)
```

```
##          CR40.2002_044w CR40.2004_044w CR40.2005_044w CR40.2006_044w
## 2              7953          18066          22056          10786
## 1239            4224          15247          5666          4349
## 186801           154           564           143           157
## 186802           6016          21858          5837          6674
## 186803           2582          10317          2453          2407
## 1506553          129           733           130           188
##          CR40.2007_044w CR40.2008_044w CR40.2010_044w AL.0004_044w AL.0005_044w
## 2              23040          15772          27846          11038          6493
## 1239            23822          11088          8162           3296          6130
## 186801           888           373           216           133           207
## 186802           35103          16897          11008          4258          10365
## 186803           14085          6776           4081          1695          4082
## 1506553           796           494           234           90           225
##          AL.0006_044w AL.0007_044w AL.0008_044w AL.0010_044w AL.0012_044w
## 2              2588          13389          7739          15706          8642
## 1239            1941          13640          10183          13891          11088
## 186801           73           462           262           508           365
## 186802           2796          21574          14860          21257          18023
## 186803           1386           7990          5573          10053          8115
## 1506553           56           523           519           504           514
##          AL.0014_044w AL.0015_044w
## 2              14922          23453
## 1239            13761           8700
## 186801           651           295
## 186802           27305          10149
## 186803           10761           3840
## 1506553           601           220
```

- 6565 taxa, 16 samples
- Make sure these numbers (especially the number of samples) make sense

Import sample metadata

```
# CHANGE ME: you can either create sample metadata from the sample names
# or import the metadata from a different file
meta.df <- data.frame(
  sample.ID=colnames(kraken.data.df))

# CHANGE ME: create new columns based on sample name
meta.df <- meta.df %>%
  tidyr::separate(sample.ID, c("diet.mouseID", "age.weeks"), sep="_", remove=F) %>%
  tidyr::separate(diet.mouseID, c("diet", "mouseID"), sep="\\.") %>%
  column_to_rownames("sample.ID")

# CHANGE ME: convert certain columns to factors in order to specify their order in plots
meta.df <- meta.df %>%
  mutate(diet.fctr=factor(diet, levels=c("AL", "CR40")))
```

Convert Consensus Lineage column into a taxonomy df

```
kraken.tax.df <- data.frame(str_split_fixed(kraken.df$Consensus.Lineage, "__|; ", n=14)[, seq(2,14,2)])
colnames(kraken.tax.df) <- c("kingdom", "phylum", "class", "order", "family", "genus", "species")
rownames(kraken.tax.df) <- kraken.df$OTU.ID
```

Create phyloseq object

```
physeq <- phyloseq(
  kraken.data.df %>% as.matrix %>% otu_table(taxa_are_rows=T),
  meta.df %>% sample_data,
  kraken.tax.df %>% as.matrix %>% tax_table)
```

Subset to bacteria

```
physeq.bac <- physeq %>% subset_taxa(kingdom=="Bacteria")
```

- 6242 bacterial taxa

Percent of reads assigned to each taxonomy level

```
physeq.bac.tax.df.w.tax.sum <- data.frame(taxa.sum=taxa_sums(physeq.bac)) %>%
  merge(data.frame(tax_table(physeq.bac)), by="row.names")
```

```
1 - sum(physeq.bac.tax.df.w.tax.sum$taxa.sum[physeq.bac.tax.df.w.tax.sum$family == ""]) / sum(physeq.bac
```

```
## [1] 0.9491975
```

```
1 - sum(physeq.bac.tax.df.w.tax.sum$taxa.sum[physeq.bac.tax.df.w.tax.sum$genus == ""]) / sum(physeq.bac
```

```
## [1] 0.9268306
```

```
1 - sum(physeq.bac.tax.df.w.tax.sum$taxa.sum[physeq.bac.tax.df.w.tax.sum$species == ""]) / sum(physeq.b
```

```
## [1] 0.8740184
```

- These numbers describe what percent of reads could be mapped to a particular taxonomy level
- For example, 92.7% of reads could be mapped at least to the genus level

Aggregate to different taxonomy levels

```
physeq.bac.phylum <- physeq.bac %>% tax_glom(taxrank="phylum")
physeq.bac.class <- physeq.bac %>% tax_glom(taxrank="class")
physeq.bac.family <- physeq.bac %>% tax_glom(taxrank="family")
physeq.bac.genus <- physeq.bac %>% tax_glom(taxrank="genus")
physeq.bac.species <- physeq.bac %>% tax_glom(taxrank="species")
```

```
ntaxa(physeq.bac.phylum)
```

```
## [1] 37
```

```
ntaxa(physeq.bac.class)
```

```
## [1] 73
ntaxa(physeq.bac.family)

## [1] 384
ntaxa(physeq.bac.genus)

## [1] 1463
ntaxa(physeq.bac.species)

## [1] 5487
```

- This tells you how many unique bacterial phyla, families, genera, etc. were detected

Add genus_and_species annotation to physeq.bac.species for plotting purposes

```
tax_table(physeq.bac.species) <- data.frame(tax_table(physeq.bac.species)) %>%
  mutate(genus_and_species=str_c(genus, species, sep=" ")) %>%
  as.matrix %>% tax_table
```

Optional: Filtration

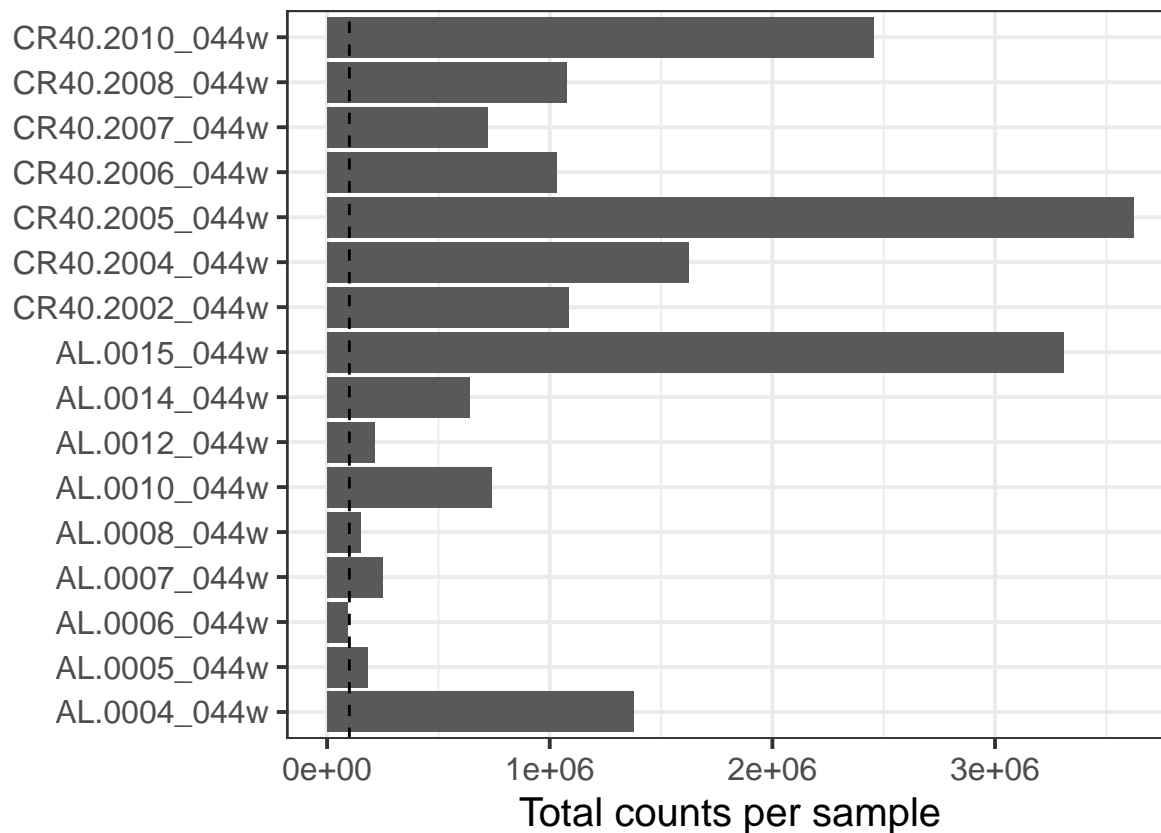
Samples

It may be helpful to discard samples with very few reads and taxons that were observed very infrequently. The defaults provided here are to include samples with at least 10k counts, and taxons with at least 0.01% relative abundance in 20% of samples.

Here, I perform filtering for just the genus phyloseq object, but could do the same thing for the phyloseq objects at other taxonomy levels.

```
# CHANGE ME: update filtering parameters as desired
MIN.COUNTS.PER.SAMPLE <- 1e5

data.frame(sample_sums(physeq.bac)) %>%
  setNames("sample.sum") %>%
  rownames_to_column("sample.ID") %>%
  ggplot(aes(x=sample.sum, y=sample.ID)) +
  geom_bar(stat="identity") +
  geom_vline(xintercept=MIN.COUNTS.PER.SAMPLE, lty=2) +
  labs(x="Total counts per sample", y="")
```



```
samples.to.keep.bool <- sample_sums(physeq.bac) > MIN.COUNTS.PER.SAMPLE

# CHANGE: apply filtering for all phyloseq objects that you want
physeq.bac.genus.filt <- physeq.bac.genus %>%
  subset_samples(samples.to.keep.bool)
physeq.bac.species.filt <- physeq.bac.species %>%
  subset_samples(samples.to.keep.bool)
```

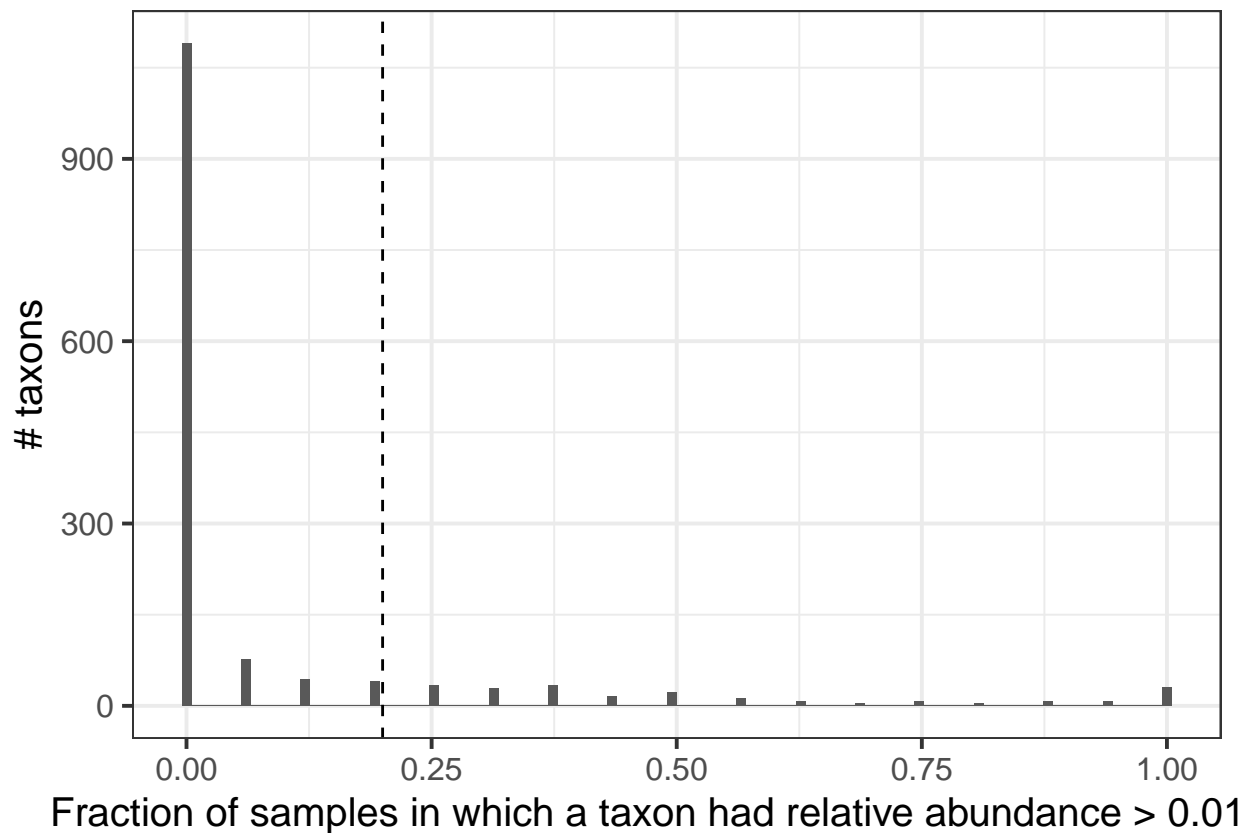
Taxa

Can filter the phyloseq objects at other taxonomy levels too.

Genus

```
# CHANGE: update filtering parameters as desired
MIN.REL.ABUND <- 1e-4
MIN.PERC.SAMPLES.W.TAXON.AT.REL.ABUND <- 0.2

bac.genus.rel.abund.mat <- physeq.bac.genus %>%
  transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%
  otu_table %>% as.matrix
data.frame(
  frac.taxon.w.sufficient.rel.abund=rowSums(bac.genus.rel.abund.mat > MIN.REL.ABUND) / ncol(bac.genus.r
  ggplot(aes(frac.taxon.w.sufficient.rel.abund)) +
  geom_histogram(bins=100) +
  geom_vline(xintercept=MIN.PERC.SAMPLES.W.TAXON.AT.REL.ABUND, lty=2) +
  labs(y="# taxa", x=sprintf("Fraction of samples in which a taxon had relative abundance > %.3f%%",
```

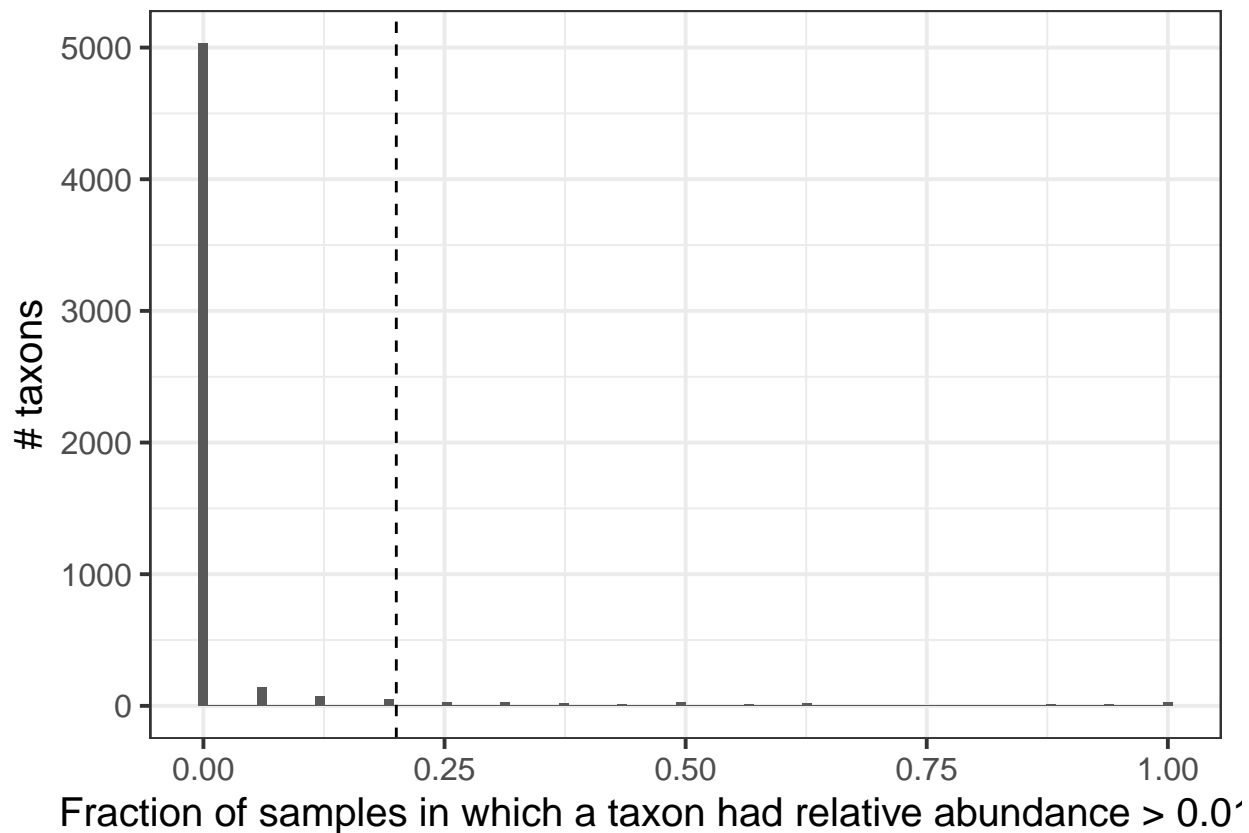


```
genera.to.keep.bool <- rowSums(bac.genus.rel.abund.mat > MIN.REL.ABUND) / ncol(bac.genus.rel.abund.mat)
physeq.bac.genus.filt <- physeq.bac.genus.filt %>%
  subset_taxa(genera.to.keep.bool)
```

Species

```
# CHANGE ME: update filtering parameters as desired
MIN.REL.ABUND <- 1e-4
MIN.PERC.SAMPLES.W.TAXON.AT.REL.ABUND <- 0.2

bac.species.rel.abund.mat <- physeq.bac.species %>%
  transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%
  otu_table %>% as.matrix
data.frame(
  frac.taxon.w.sufficient.rel.abund=rowSums(bac.species.rel.abund.mat > MIN.REL.ABUND) / ncol(bac.species.rel.abund.mat)
) %>%
  ggplot(aes(frac.taxon.w.sufficient.rel.abund)) +
  geom_histogram(bins=100) +
  geom_vline(xintercept=MIN.PERC.SAMPLES.W.TAXON.AT.REL.ABUND, lty=2) +
  labs(y="# taxa", x=sprintf("Fraction of samples in which a taxon had relative abundance > %.3f%", MIN.PERC.SAMPLES.W.TAXON.AT.REL.ABUND))
```



```
species.to.keep.bool <- rowSums(bac.species.rel.abund.mat > MIN.REL.ABUND) / ncol(bac.species.rel.abund.mat)
physeq.bac.species.filt <- physeq.bac.species.filt %>%
  subset_taxa(species.to.keep.bool)
```

Summary of filtration

physeq.bac.genus

```
## phyloseq-class experiment-level object
## otu_table() OTU Table: [ 1463 taxa and 16 samples ]:
## sample_data() Sample Data: [ 16 samples by 4 sample variables ]:
## tax_table() Taxonomy Table: [ 1463 taxa by 7 taxonomic ranks ]:
## taxa are rows
```

physeq.bac.genus.filt

```
## phyloseq-class experiment-level object
## otu_table() OTU Table: [ 213 taxa and 15 samples ]:
## sample_data() Sample Data: [ 15 samples by 4 sample variables ]:
## tax_table() Taxonomy Table: [ 213 taxa by 7 taxonomic ranks ]:
## taxa are rows
```

physeq.bac.species

```
## phyloseq-class experiment-level object
## otu_table() OTU Table: [ 5487 taxa and 16 samples ]:
## sample_data() Sample Data: [ 16 samples by 4 sample variables ]:
## tax_table() Taxonomy Table: [ 5487 taxa by 8 taxonomic ranks ]:
## taxa are rows
```

```
physeq.bac.species.filt
```

```
## phyloseq-class experiment-level object
## otu_table() OTU Table: [ 197 taxa and 15 samples ]:
## sample_data() Sample Data: [ 15 samples by 4 sample variables ]:
## tax_table() Taxonomy Table: [ 197 taxa by 8 taxonomic ranks ]:
## taxa are rows

# If you choose not to perform filtering, run the following lines:
# physeq.bac.genus.filt <- physeq.bac.genus
# physeq.bac.species.filt <- physeq.bac.species
```

- 16 samples -> 15 samples
- 1463 genera -> 213 genera
- 5487 species -> 197 species

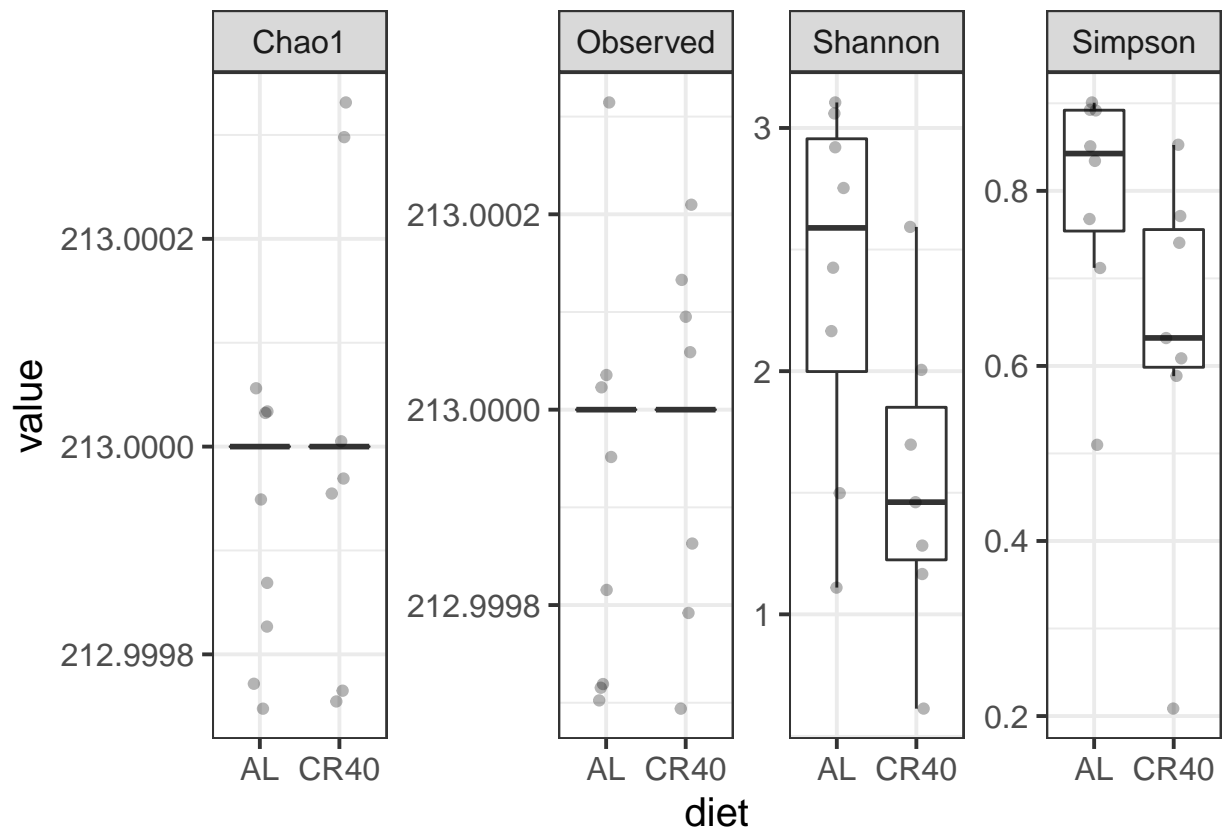
Alpha diversity

Genus

```
alpha.div.genus.df <- estimate_richness(
  physeq.bac.genus.filt, measures=c("Observed", "Chao1", "Shannon", "Simpson"))

alpha.div.genus.df %>%
  merge(meta.df, by="row.names") %>%
  pivot_longer(c(Observed, Chao1, Shannon, Simpson), names_to="diversity") %>%

# CHANGE ME to show desired comparisons
ggplot(aes(x=diet, y=value)) +
  geom_boxplot(outlier.shape=NA) +
  geom_jitter(width=0.1, alpha=0.3) +
  facet_wrap(~diversity, ncol=4, scales="free_y")
```

After rarefaction

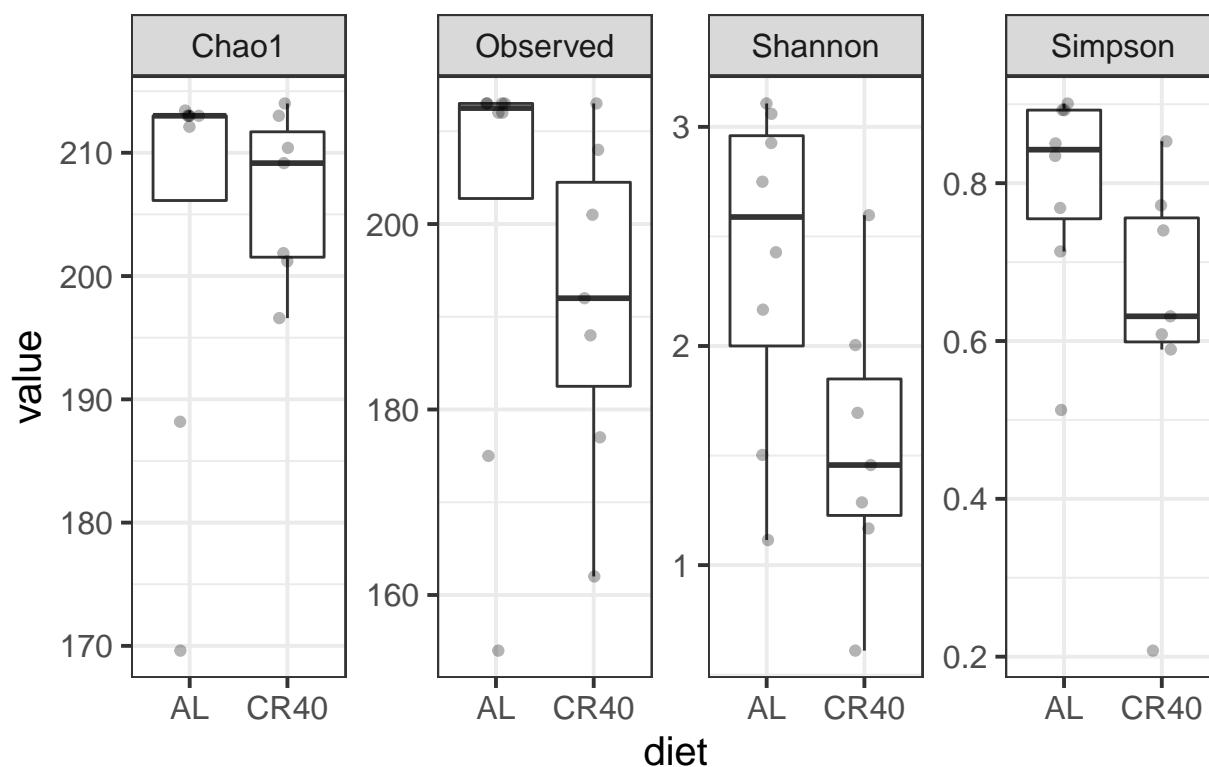
Alpha diversity is sensitive to sequencing depth, so I recommend performing rarefaction prior to running alpha diversity analysis so that the results aren't confounded by sequencing depth.

```
alpha.div.genus.rarefied.df <- physeq.bac.genus.filt %>%
  rarefy_even_depth(rngseed=101, replace=F, verbose=F) %>%
  estimate_richness(measures=c("Observed", "Chao1", "Shannon", "Simpson"))

alpha.div.genus.rarefied.df %>%
  merge(meta.df, by="row.names") %>%
  pivot_longer(c(Observed, Chao1, Shannon, Simpson), names_to="diversity") %>%

  # CHANGE ME to show desired comparisons
  ggplot(aes(x=diet, y=value)) +
  geom_boxplot(outlier.shape=NA) +
  geom_jitter(width=0.1, alpha=0.3) +
  facet_wrap(~diversity, ncol=4, scales="free_y") +
  labs(title="After rarefying")
```

After rarefying

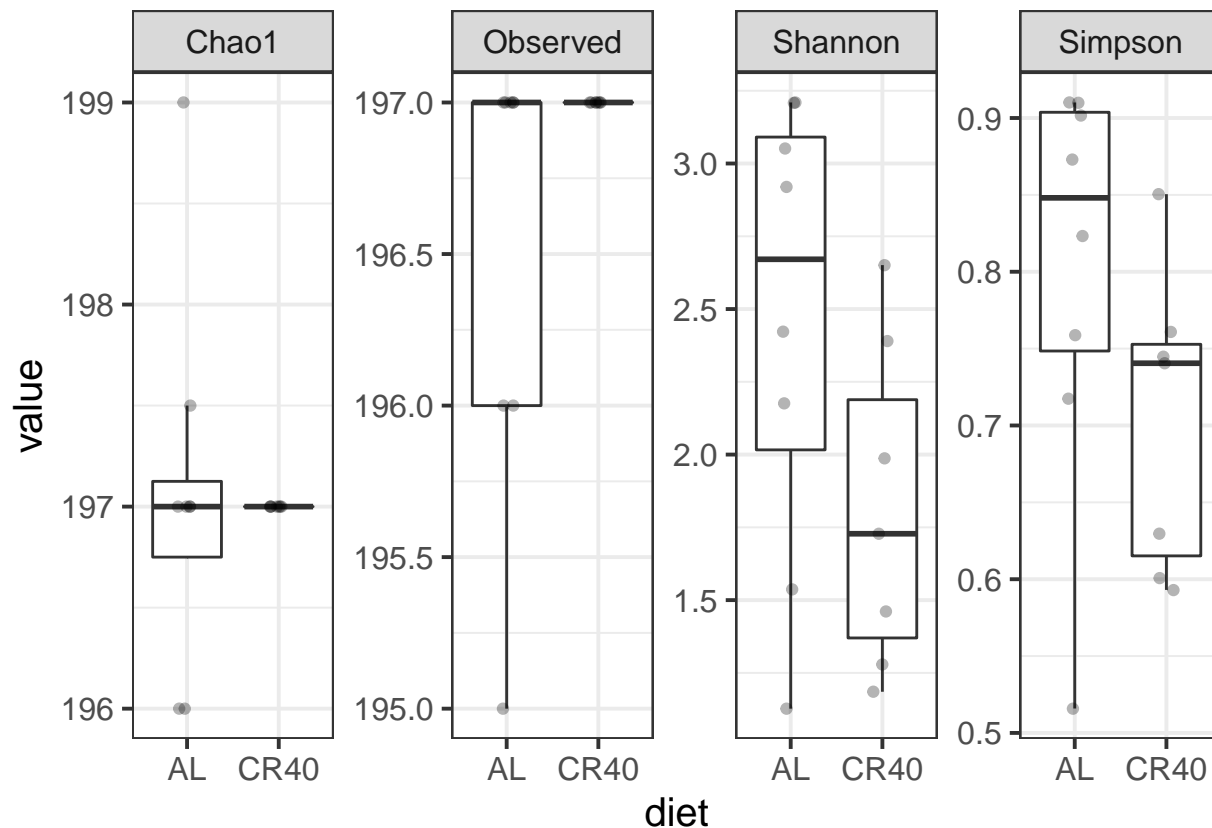


Species

```
alpha.div.species.df <- estimate_richness(
  physeq.bac.species.filt, measures=c("Observed", "Chao1", "Shannon", "Simpson"))
```

```
alpha.div.species.df %>%
  merge(meta.df, by="row.names") %>%
  pivot_longer(c(Observed, Chao1, Shannon, Simpson), names_to="diversity") %>%
```

```
# CHANGE ME to show desired comparisons
ggplot(aes(x=diet, y=value)) +
  geom_boxplot(outlier.shape=NA) +
  geom_jitter(width=0.1, alpha=0.3) +
  facet_wrap(~diversity, ncol=4, scales="free_y")
```



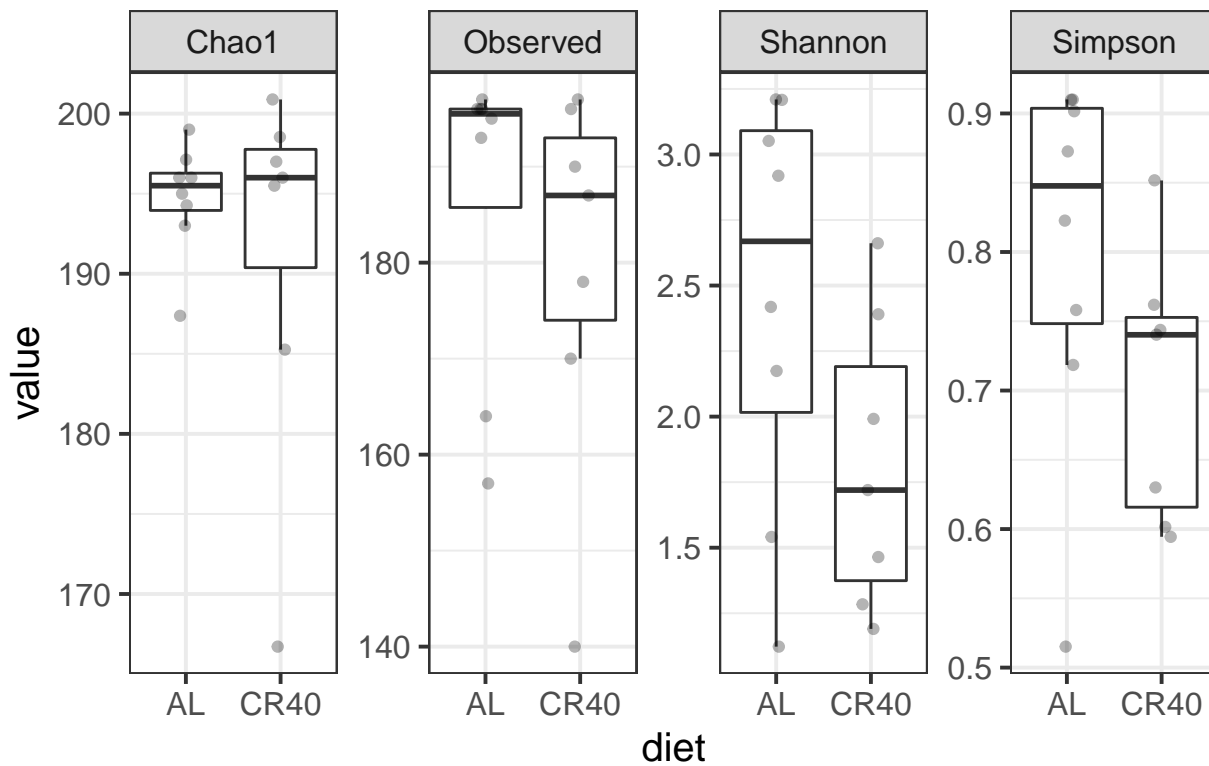
After rarefaction

```
alpha.div.species.rarefied.df <- physeq.bac.species.filt %>%
  rarefy_even_depth(rngseed=101, replace=F, verbose=F) %>%
  estimate_richness(measures=c("Observed", "Chao1", "Shannon", "Simpson"))

alpha.div.species.rarefied.df %>%
  merge(meta.df, by="row.names") %>%
  pivot_longer(c(Observed, Chao1, Shannon, Simpson), names_to="diversity") %>%

  # CHANGE ME to show desired comparisons
  ggplot(aes(x=diet, y=value)) +
  geom_boxplot(outlier.shape=NA) +
  geom_jitter(width=0.1, alpha=0.3) +
  facet_wrap(~diversity, ncol=4, scales="free_y") +
  labs(title="After rarefying")
```

After rarefying



Barplots

Phylum

```
top.n.phyla <- data.frame(tax_table(physeq.bac.phylum)) [
  names(sort(taxa_sums(physeq.bac.phylum), decreasing=T))[1:9], "phylum"]

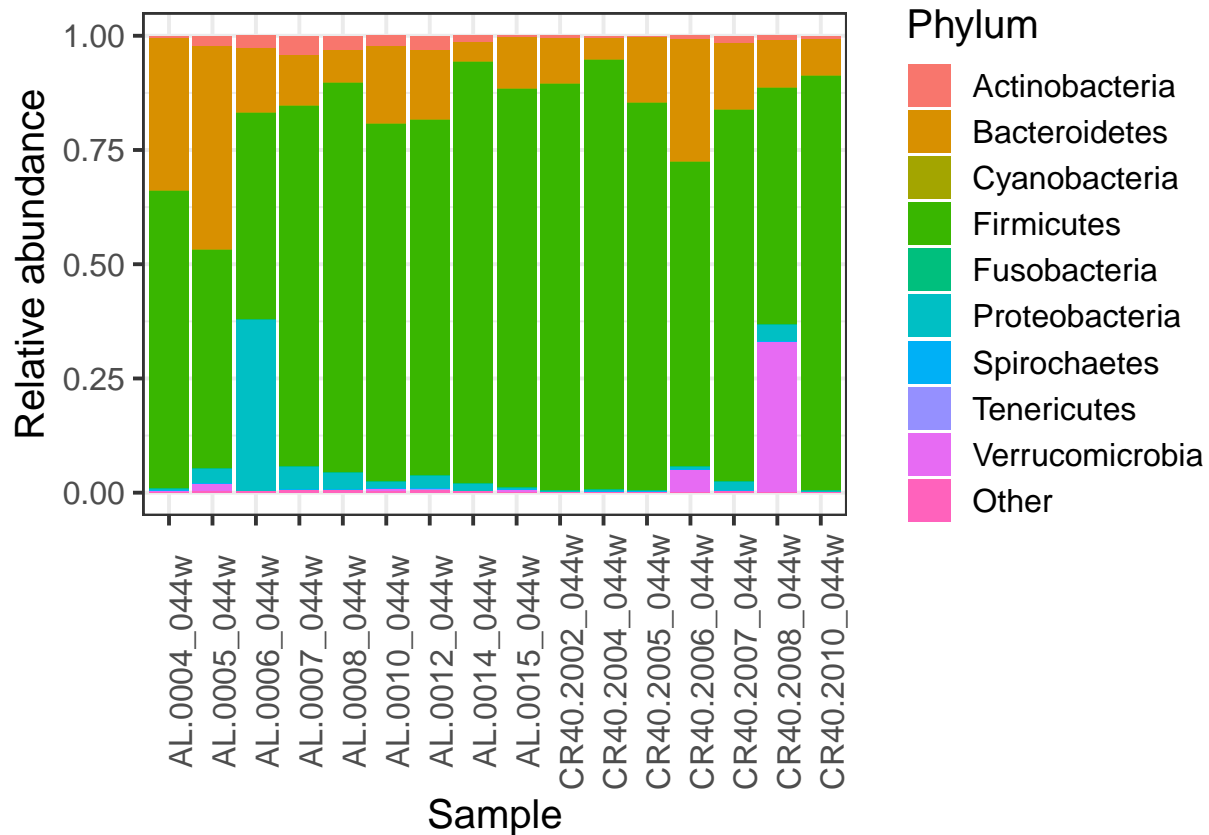
physeq.bac.phylum %>%

  # compute relative abundance
  transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%

  # convert to df
  psmelt %>%

  # aggregate less common taxa into "Other"
  mutate(agg.phylum=fct_relevel(
    ifelse(phylum %in% top.n.phyla, phylum, "Other"), "Other", after=Inf)) %>%

  # CHANGEME to show desired comparisons
  ggplot(aes(x=Sample, y=Abundance, fill=agg.phylum)) +
  geom_bar(stat="identity") +
  labs(y="Relative abundance", fill="Phylum") +
  theme(axis.text.x=element_text(angle=90))
```



Genus

```
top.n.genera <- data.frame(tax_table(physeq.bac.genus.filt))[
  names(sort(taxa_sums(physeq.bac.genus.filt), decreasing=T))[1:9], "genus"]

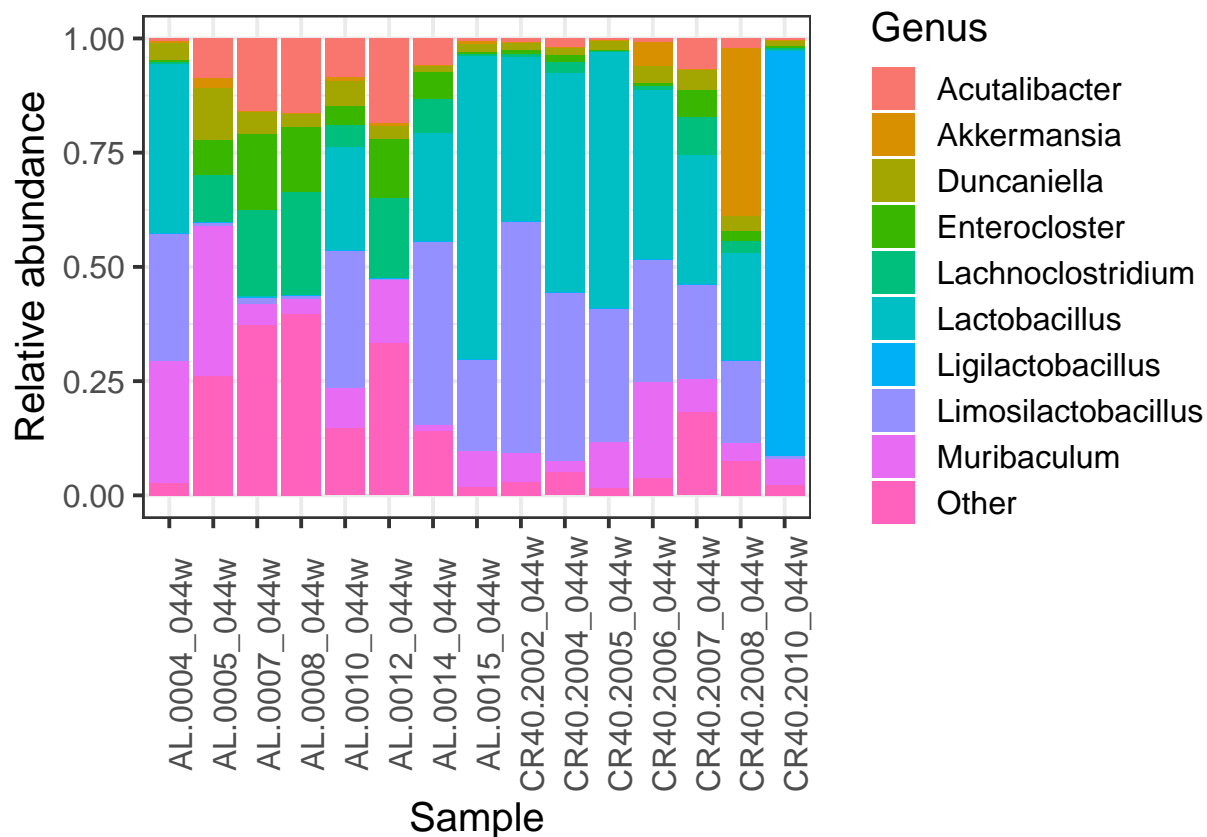
physeq.bac.genus.filt %>%

  # compute relative abundance
  transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%

  # convert to df
  psmelt %>%

  # aggregate less common taxa into "Other"
  mutate(agg.genus=fct_relevel(
    ifelse(genus %in% top.n.genera, genus, "Other"), "Other", after=Inf)) %>%

  # CHANGEME to show desired comparisons
  ggplot(aes(x=Sample, y=Abundance, fill=agg.genus)) +
  geom_bar(stat="identity") +
  labs(y="Relative abundance", fill="Genus") +
  theme(axis.text.x=element_text(angle=90))
```



Species

Use `genus_and_species` annotation to improve the labels on the plot.

```
top.n.species <- data.frame(tax_table(physeq.bac.species.filt))[
  names(sort(taxa_sums(physeq.bac.species.filt), decreasing=T))[1:9], "genus_and_species"]

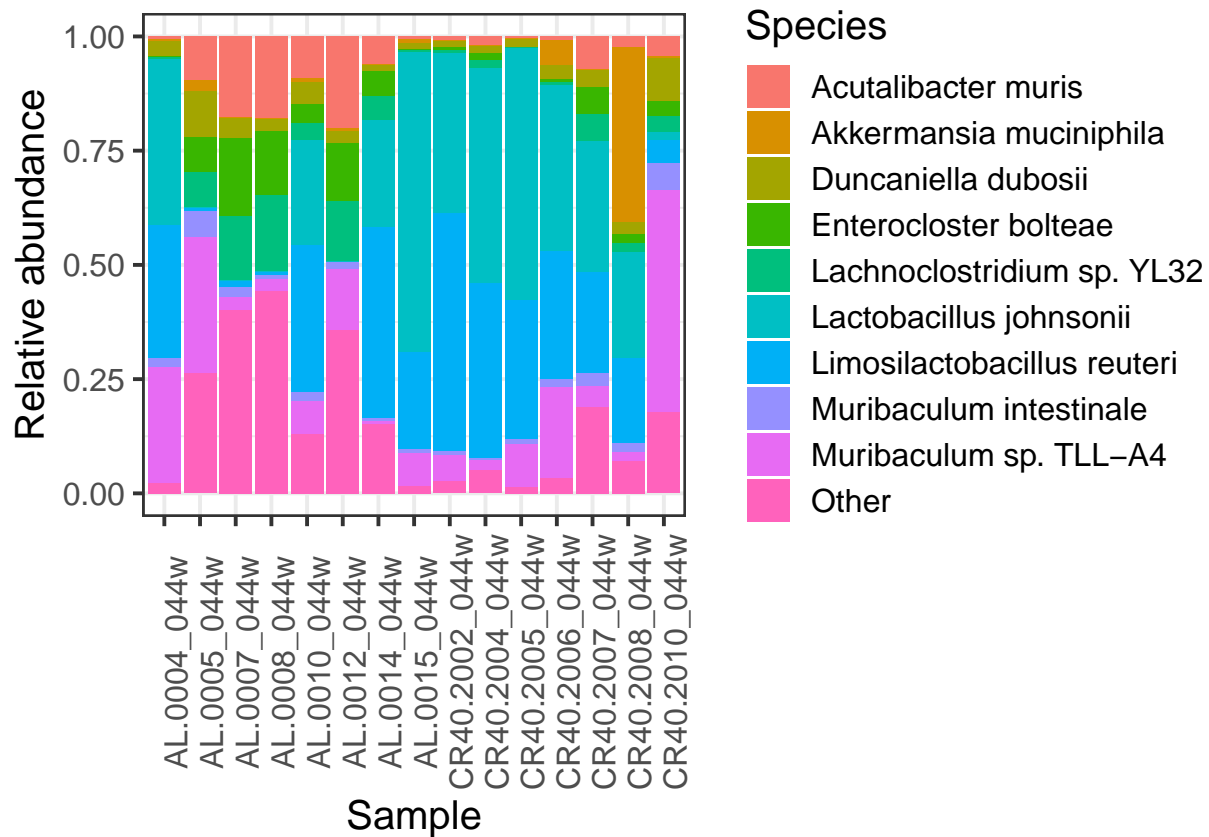
physeq.bac.species.filt %>%

  # compute relative abundance
  transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%

  # convert to df
  psmelt %>%

  # aggregate less common taxa into "Other"
  mutate(agg.species=fct_relevel(
    ifelse(genus_and_species %in% top.n.species, genus_and_species, "Other"), "Other", after=Inf)) %>%

  # CHANGEME to show desired comparisons
  ggplot(aes(x=Sample, y=Abundance, fill=agg.species)) +
  geom_bar(stat="identity") +
  labs(y="Relative abundance", fill="Species") +
  theme(axis.text.x=element_text(angle=90))
```



PCoA

Genus

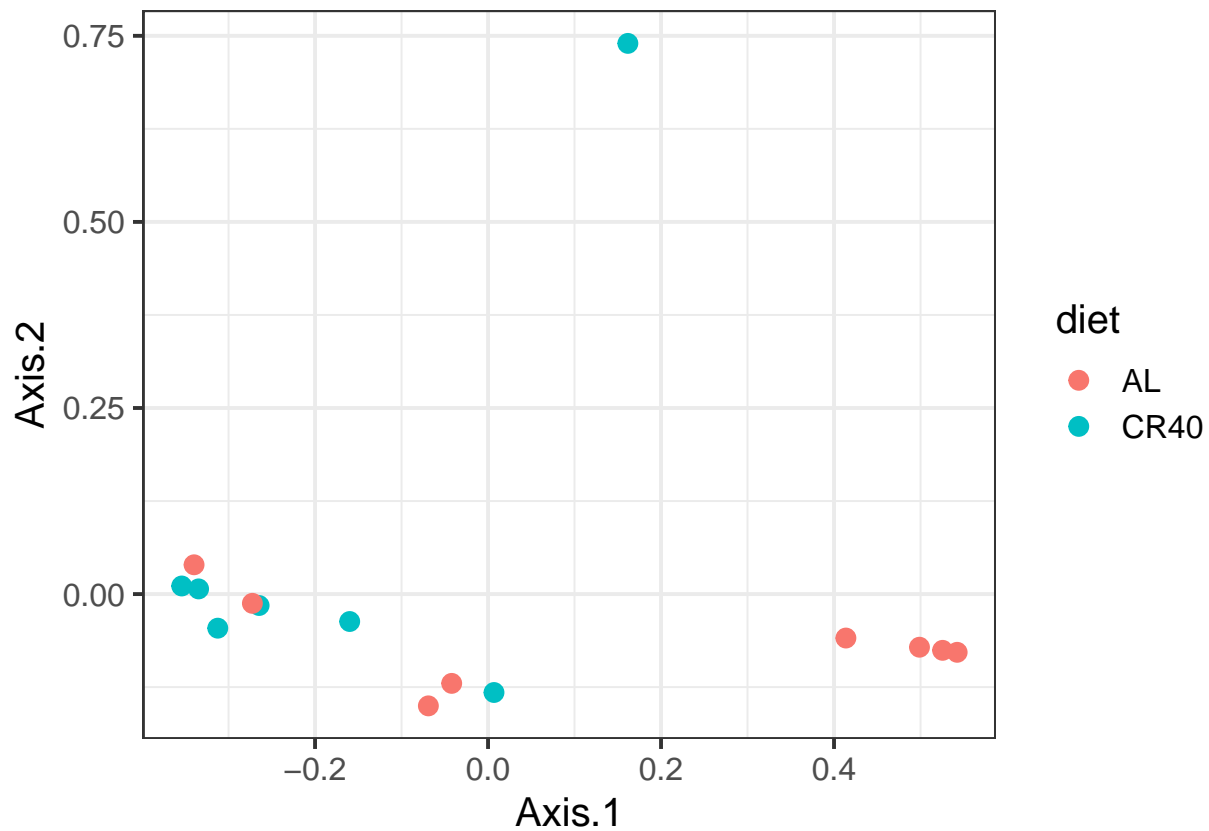
```
pcoa.bac.genus <- physeq.bac.genus.filt %>%

# compute relative abundance
transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%

# perform PCoA using Bray-Curtis distance
ordinate(method="MDS", distance="bray")

# create coordinates
pcoa.bac.genus.df <- plot_ordination(physeq.bac.genus.filt, pcoa.bac.genus,
                                   type="samples", justDF=T)

# CHANGEME to show desired comparisons
pcoa.bac.genus.df %>%
  ggplot(aes(x=Axis.1, y=Axis.2, color=diet)) +
  geom_point(size=3)
```



Species

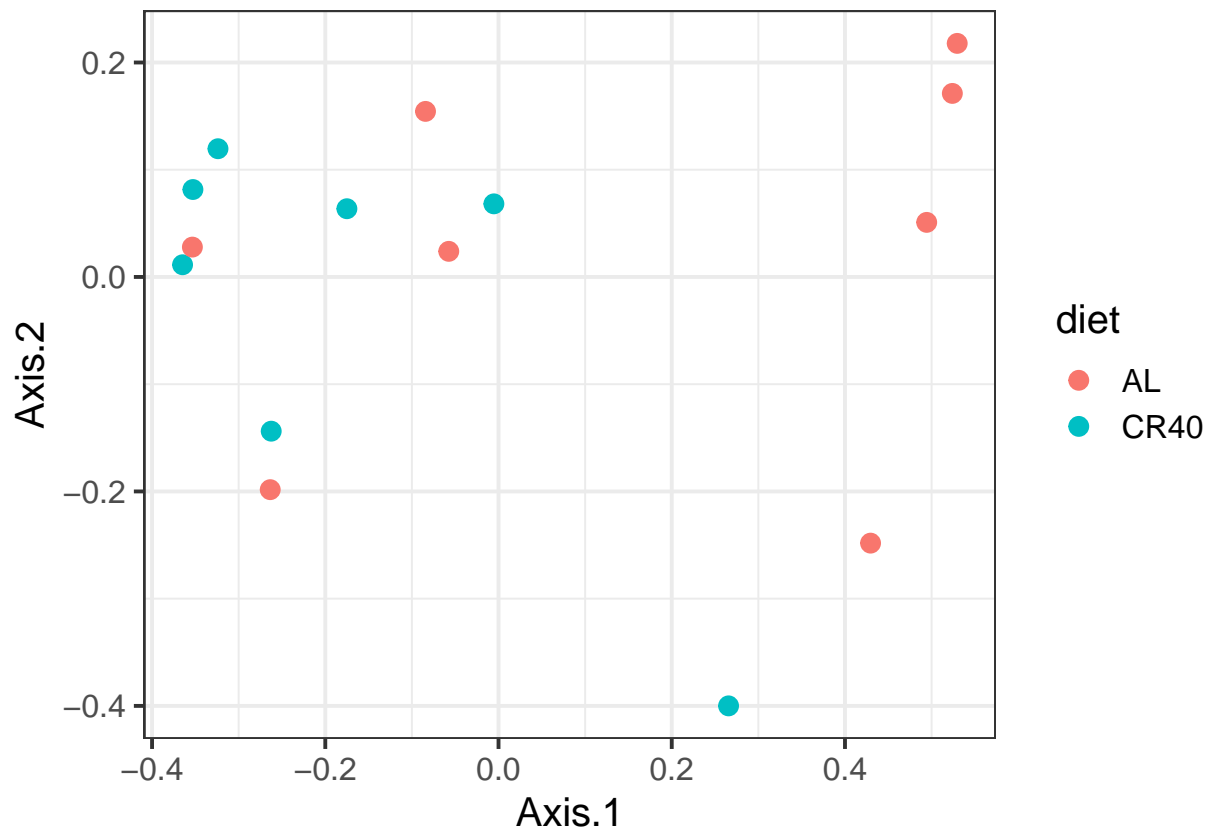
```
pcoa.bac.species <- physeq.bac.species.filt %>%

# compute relative abundance
transform_sample_counts(function(OTU) OTU/sum(OTU)) %>%

# perform PCoA using Bray-Curtis distance
ordinate(method="MDS", distance="bray")

# create coordinates
pcoa.bac.species.df <- plot_ordination(physeq.bac.species.filt, pcoa.bac.species,
                                     type="samples", justDF=T)

# CHANGE ME to show desired comparisons
pcoa.bac.species.df %>%
  ggplot(aes(x=Axis.1, y=Axis.2, color=diet)) +
  geom_point(size=3)
```

DESeq2 to find differential taxa

Genus

```
deseq.genus <- physeq.bac.genus.filt %>%

# CHANGE: update the design formula to perform desired comparisons
phyloseq_to_deseq2(~diet)

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

deseq.genus <- DESeq(deseq.genus)

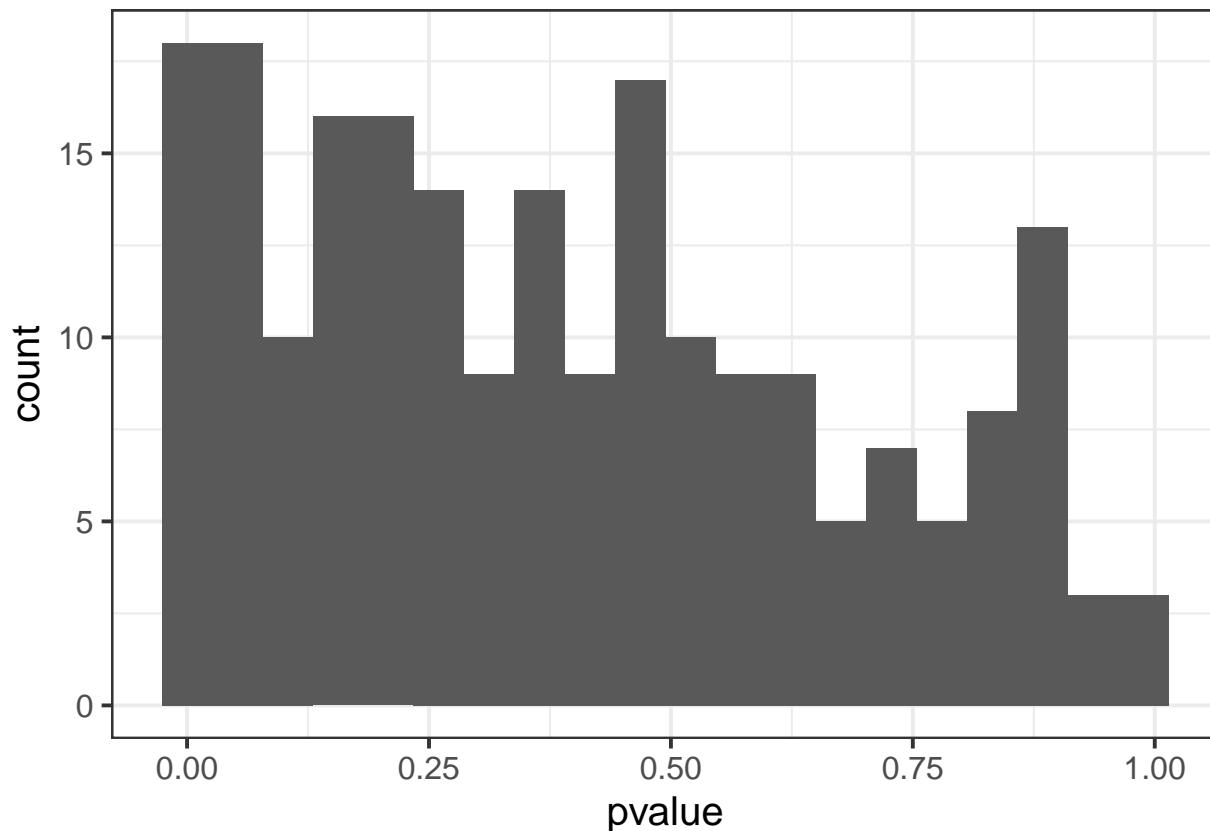
## estimating size factors
## estimating dispersions
## gene-wise dispersion estimates
## mean-dispersion relationship
## -- note: fitType='parametric', but the dispersion trend was not well captured by the
## function: y = a/x + b, and a local regression fit was automatically substituted.
## specify fitType='local' or 'mean' to avoid this message next time.

## final dispersion estimates
## fitting model and testing
```

```
## -- replacing outliers and refitting for 4 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions
## fitting model and testing
```

```
results(deseq.genus) %>% as.data.frame %>%
  ggplot(aes(pvalue)) +
  geom_histogram(bins=20)
```



- The shape of this histogram tells you whether to expect any significant results

```
results(deseq.genus) %>% as.data.frame %>%
  dplyr::filter(padj < 0.1)
```

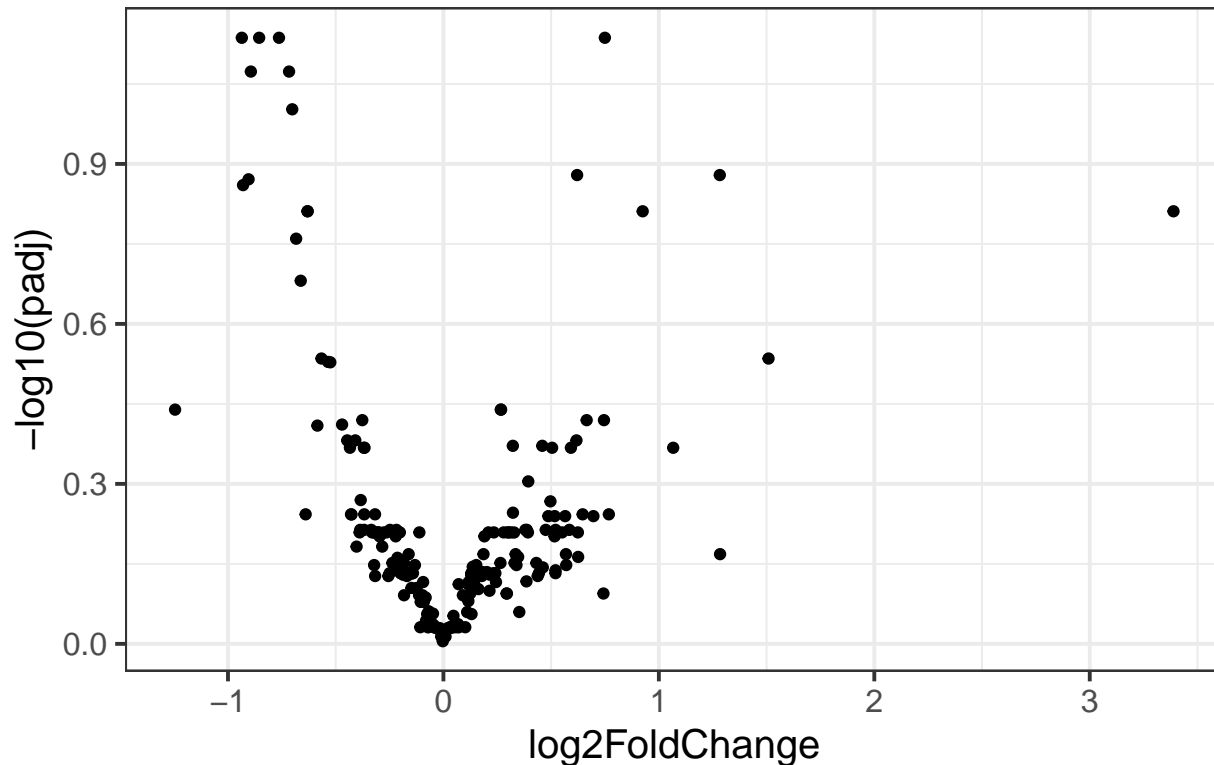
##	baseMean	log2FoldChange	lfcSE	stat	pvalue	padj
## 39488	393.97495	-0.8936661	0.2941429	-3.038204	0.0023799284	0.08448746
## 2763056	1450.57113	-0.7016046	0.2385345	-2.941313	0.0032682398	0.09944787
## 35701	17.61384	-0.7630718	0.2356969	-3.237514	0.0012057621	0.07301969
## 2583452	46.21434	-0.7164464	0.2347089	-3.052489	0.0022695182	0.08448746
## 365349	23.02017	0.7494517	0.2243861	3.340009	0.0008377557	0.07301969
## 128785	15.30414	-0.8552258	0.2431065	-3.517907	0.0004349655	0.07301969
## 81412	13.61889	-0.9356860	0.2923441	-3.200633	0.0013712618	0.07301969

- 7 results with adjusted p-value < 0.1
- The adjusted p-value threshold is somewhat arbitrary

```
results(deseq.genus) %>% as.data.frame %>%
  ggplot(aes(x=log2FoldChange, y=-log10(padj))) +
```

```
geom_point() +
labs(title="Volcano plot, genera")
```

Volcano plot, genera



Species

```
deseq.species <- physeq.bac.species.filt %>%

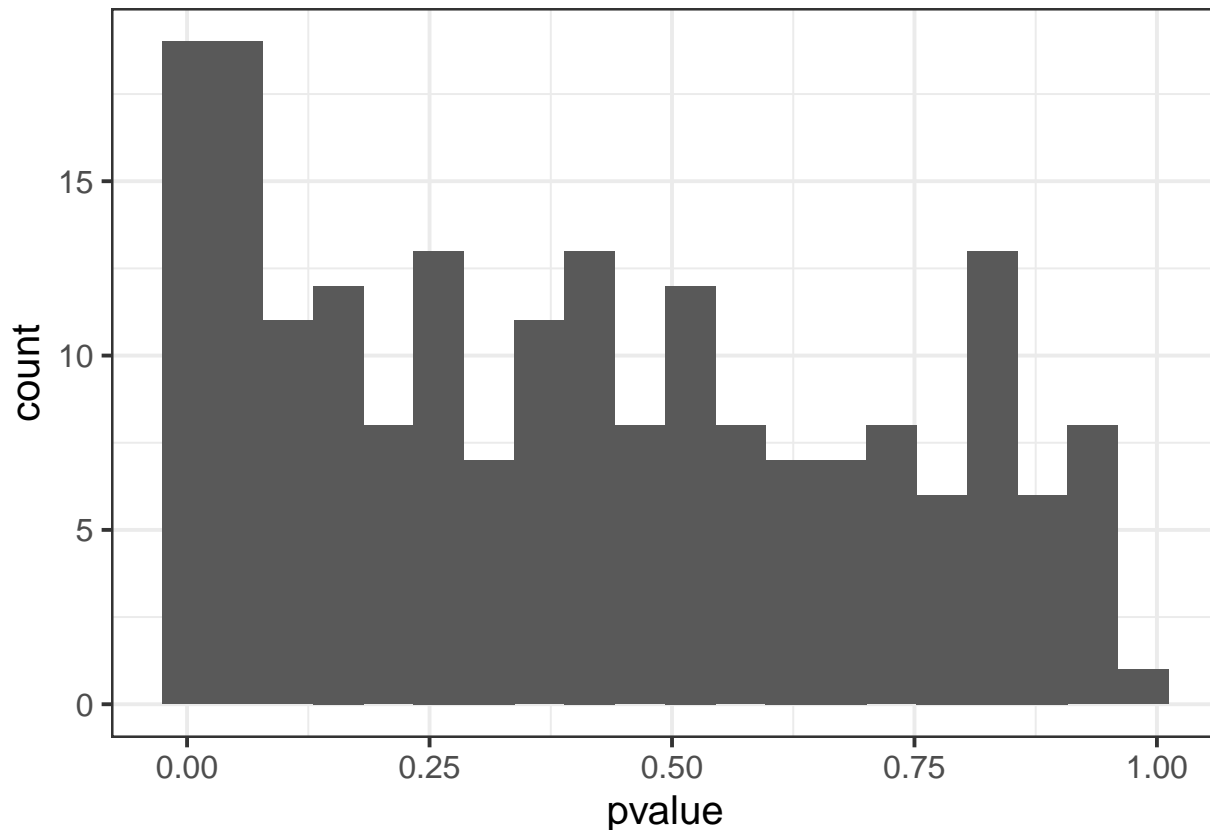
# CHANGE ME: update the design formula to perform desired comparisons
phyloseq_to_deseq2(~diet)

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
deseq.species <- DESeq(deseq.species)

## estimating size factors
## estimating dispersions
## gene-wise dispersion estimates
## mean-dispersion relationship
## -- note: fitType='parametric', but the dispersion trend was not well captured by the
## function: y = a/x + b, and a local regression fit was automatically substituted.
## specify fitType='local' or 'mean' to avoid this message next time.
## final dispersion estimates
```

```
## fitting model and testing
## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
## estimating dispersions
## fitting model and testing
results(deseq.species) %>% as.data.frame %>%
  ggplot(aes(pvalue)) +
  geom_histogram(bins=20)
```

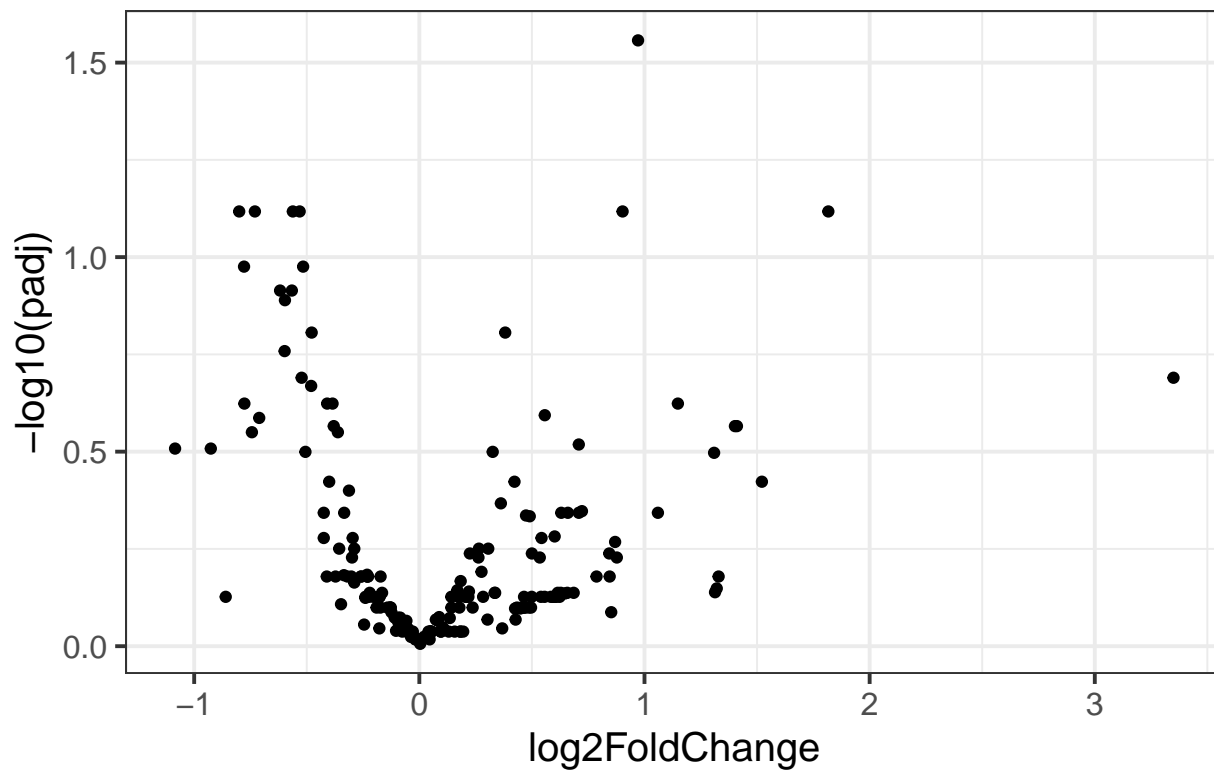


```
results(deseq.species) %>% as.data.frame %>%
  dplyr::filter(padj < 0.1)
```

##	baseMean	log2FoldChange	lfcSE	stat	pvalue	padj
## 1912897	679.75388	0.9713758	0.2551635	3.806877	0.0001407329	0.02772438
## 39488	375.83986	-0.7303005	0.2433224	-3.001370	0.0026876759	0.07631305
## 351091	386.04652	-0.5315480	0.1770845	-3.001662	0.0026850977	0.07631305
## 1584	36.09639	0.9027695	0.2943759	3.066723	0.0021641909	0.07631305
## 414771	93.79130	1.8166514	0.5773366	3.146607	0.0016517677	0.07631305
## 1685	72.10095	-0.5618419	0.1873638	-2.998667	0.0027116313	0.07631305
## 128785	13.34972	-0.8000896	0.2445831	-3.271239	0.0010707740	0.07631305

```
results(deseq.species) %>% as.data.frame %>%
  ggplot(aes(x=log2FoldChange, y=-log10(padj))) +
  geom_point() +
  labs(title="Volcano plot, species")
```

Volcano plot, species



Save for reproducibility

```
Sys.Date()
```

```
## [1] "2022-12-15"
```

```
sessionInfo()
```

```
## R version 4.2.0 (2022-04-22)
```

```
## Platform: x86_64-apple-darwin17.0 (64-bit)
```

```
## Running under: macOS Big Sur/Monterey 10.16
```

```
##
```

```
## Matrix products: default
```

```
## BLAS: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dylib
```

```
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dylib
```

```
##
```

```
## locale:
```

```
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
##
```

```
## attached base packages:
```

```
## [1] stats4 stats graphics grDevices utils datasets methods
```

```
## [8] base
```

```
##
```

```
## other attached packages:
```

```
## [1] speedyseq_0.5.3.9018 DESeq2_1.36.0
```

```
## [3] SummarizedExperiment_1.26.1 Biobase_2.56.0
```

```
## [5] MatrixGenerics_1.8.1 matrixStats_0.62.0
```

```

## [7] GenomicRanges_1.48.0      GenomeInfoDb_1.32.4
## [9] IRanges_2.30.1             S4Vectors_0.34.0
## [11] BiocGenerics_0.42.0        phyloseq_1.40.0
## [13] forcats_0.5.2              stringr_1.4.1
## [15] dplyr_1.0.10               purrr_0.3.4
## [17] readr_2.1.2                tidyr_1.2.1
## [19] tibble_3.1.8               ggplot2_3.3.6
## [21] tidyverse_1.3.2
##
## loaded via a namespace (and not attached):
## [1] googledrive_2.0.0          colorspace_2.0-3          ellipsis_0.3.2
## [4] XVector_0.36.0            fs_1.5.2                 rstudioapi_0.14
## [7] farver_2.1.1              bit64_4.0.5              AnnotationDbi_1.58.0
## [10] fansi_1.0.3               lubridate_1.8.0          xml2_1.3.3
## [13] codetools_0.2-18          splines_4.2.0            cachem_1.0.6
## [16] geneplotter_1.74.0        knitr_1.40               ade4_1.7-19
## [19] jsonlite_1.8.3           broom_1.0.1             annotate_1.74.0
## [22] cluster_2.1.4            dbplyr_2.2.1            png_0.1-7
## [25] compiler_4.2.0           httr_1.4.4              backports_1.4.1
## [28] assertthat_0.2.1         Matrix_1.5-1            fastmap_1.1.0
## [31] gargle_1.2.1             cli_3.4.1               htmltools_0.5.3
## [34] tools_4.2.0              igraph_1.3.4            gtable_0.3.1
## [37] glue_1.6.2               GenomeInfoDbData_1.2.8  reshape2_1.4.4
## [40] Rcpp_1.0.9               cellranger_1.1.0        vctrs_0.5.0
## [43] Biostrings_2.64.1        rhdf5filters_1.8.0      multtest_2.52.0
## [46] ape_5.6-2                nlme_3.1-159            iterators_1.0.14
## [49] xfun_0.34                rvest_1.0.3             lifecycle_1.0.3
## [52] XML_3.99-0.10           googlesheets4_1.0.1     zlibbioc_1.42.0
## [55] MASS_7.3-58.1           scales_1.2.1            hms_1.1.2
## [58] parallel_4.2.0          biomformat_1.24.0       rhdf5_2.40.0
## [61] RColorBrewer_1.1-3      yaml_2.3.6              memoise_2.0.1
## [64] RSQLite_2.2.17          stringi_1.7.8           highr_0.9
## [67] genefilter_1.78.0       foreach_1.5.2           permute_0.9-7
## [70] BiocParallel_1.30.3     rlang_1.0.6             pkgconfig_2.0.3
## [73] bitops_1.0-7            evaluate_0.17           lattice_0.20-45
## [76] Rhdf5lib_1.18.2         labeling_0.4.2          bit_4.0.4
## [79] tidyselect_1.1.2        plyr_1.8.7              magrittr_2.0.3
## [82] R6_2.5.1                generics_0.1.3          DelayedArray_0.22.0
## [85] DBI_1.1.3               pillar_1.8.1            haven_2.5.1
## [88] withr_2.5.0             mgcv_1.8-40             KEGGREST_1.36.3
## [91] survival_3.4-0          RCurl_1.98-1.8          modelr_0.1.9
## [94] crayon_1.5.1            utf8_1.2.2              tzdb_0.3.0
## [97] rmarkdown_2.17          locfit_1.5-9.6          grid_4.2.0
## [100] readxl_1.4.1            data.table_1.14.4       blob_1.2.3
## [103] vegan_2.6-2             reprex_2.0.2            digest_0.6.30
## [106] xtable_1.8-4            munsell_0.5.0

```